Object-Oriented Modelling of a Controlled Chemical Process

Nilsson, Bernt

1989

Document Version:
Publisher's PDF, also known as Version of record

CODEN: LUTFD2/(TFRT-7428)/1-6/(1989)

Object-oriented Modelling of a Controlled Chemical Process

Bernt Nilsson

Department of Automatic Control
Lund Institute of Technology
August 1989

Department of Automatic Control
Lund Institute of Technology
P.O. Box 118
S-221 00 Lund Sweden

Document name
INTERNAL REPORT

Date of issue
August 1989

Document Number
CODEN: LUTFRT/(TFRT-7428)/1-6/(1989)

Author(s)
Bernt Nilsson

Supervisor

Sponsoring organisation

Title and subtitle
Objet-oriented Modelling of a Controlled Chemical Process

Abstract

In this paper we discuss the benefits using an object-oriented approach to modelling. Models and modelling are gaining more and more interest in process industry due to the development of knowledge-based systems. The basic elements in object-oriented modelling methodology are modularization, model encapsulation, hierarchical submodel decomposition, model parameterization and inheritance. Models have an internal structure of model components, like terminals, parameters and behaviour descriptions. The model behavior can be described with equations or as a structure of submodels. Models and model components are represented as objects in single inheritance object class hierarchies. Chemical processes and control systems can be described with the same basic concepts. Tasks that are facilitated in object-oriented modelling are model reuse, model development, model refinement and model maintenance. They are all of major importance for multi purpose process modelling.

Key words
Computer simulation; computer-aided design; modeling; process control; process models; software tools.

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title

ISBN

Language
English

Number of pages
6

Recipient's notes

Security classification

Object-oriented Modelling of a Controlled Chemical Process

Bernt Nilsson
Department of Automatic Control, Lund Institute of Technology
Box 118, S-221 00 Lund, Sweden
Phone +46 46 108796, Usenet: Bernt@Control.LTH.Se

Abstract. In this paper we discuss the benefits using an object-oriented approach to modelling. Models and modelling are gaining more and more interest in process industry due to the development of knowledge-based systems. The basic elements in object-oriented modelling methodology are modularization, model encapsulation, hierarchical submodel decomposition, model parameterization and inheritance. Models have an internal structure of model components, like terminals, parameters and behaviour descriptions. The model behavior can be described with equations or as a structure of submodels. Models and model components are represented as objects in single inheritance object class hierarchies. Chemical processes and control systems can be described with the same basic concepts. Tasks that are facilitated in object-oriented modelling are model reuse, model development, model refinement and model maintenance. They are all of major importance for multi purpose process modelling.

Keywords: Computer simulation; computer-aided design; modeling; process control; process models; software tools.

1. Introduction

Models and modelling are gaining more and more interest in industry due to the development of knowledge-based systems. Models have been used in engineering for a long time. The development in computer technology makes it possible to use advanced models in industrial applications. Models are now also being used in simulation for education and training, decision support and for process operation supervision. This means that models are used in a wide range of applications, from process design to process operation. Models have therefore a long life time and are used by many different users for many different applications.

At the department of Automatic Control the project "Computer Aided Control Engineering" (CACE) has been working with new programming tools. In this project a new environment for system engineering (SEE) is under development. The basic design of the SEE prototype is composed of a model database, model/user interface and tools that operate on models. One tool is model development tool, another is a simulator tool. This architecture allows an object-oriented approach to model development and an equation-oriented approach to the problem solving. A short presentation of the SEE project is done in Nilsson et al (1989).

In this paper we are going to discuss the benefits using an object-oriented approach to modelling. Desired properties in modelling are possibilities to develop and reuse models in a modularized way. It should be possible to adapt and refine models to capture new conditions. Model maintenance becomes important when models are used for a long time.

Model structuring concepts are the key to create a modelling environment with desired properties. The models have a given internal structure of model components. An object-oriented approach to modelling represents both models and model components as objects. Modu-

larization, decomposition, parameterization and inheritance are the basic elements in an object-oriented modelling methodology.

The tools can translate the model representation into a set of equations which can be used by different algorithms.

This paper is organized as follows: An example of a process model is discussed in the next section. Object-oriented modelling and model structuring concepts are introduced in Section 3. The control system description is done in Section 4 and in Section 5 the experiences of using this kind of object-oriented modelling methodology is discussed.

2. The Tank Reactor Example

The main ideas in this work are illustrated on a minor chemical process part, namely a exothermic continuous stirred tank reactor. The reactor is assumed to be homogeneous in concentration and temperature. A chemical reaction is assumed to occur, $A \rightarrow B$, and it produces heat. The reactor vessel can now be modelled with a dynamic mass balance, dynamic component mass balances and an energy balance.

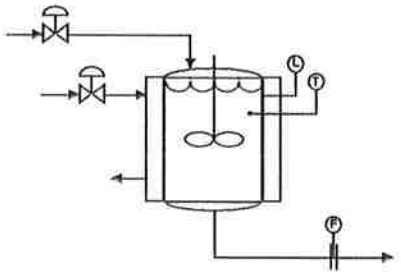

Figure 1. The continuous stirred tank reactor.

The feed to the reactor is controlled by a valve. The outflow of the reactor is set by the surrounding system.

Heat is removed by a cooling jacket. The cooling jacket can be assumed to be homogeneous and modelled by a dynamic energy balance. The cooling medium flow is also controlled by a valve.

The mathematical model of the reactor system then becomes a set of nonlinear differential equations:

$$\rho \frac{dV}{dt} = \rho q_{in} - \rho q_{out}$$

$$\frac{d(Vc)}{dt} = q_{in}c_{in} - q_{out}c + Vr$$

$$\rho C_p \frac{d(VT)}{dt} = \rho C_p q_{in} T_{in} - \rho C_p q_{out} T - Q$$

$$\rho_j C_{p_j} \frac{d(V_j T_j)}{dt} = \rho_j C_{p_j} q_j T_{j_{in}} - \rho_j C_{p_j} q_j T_j + Q$$

$$r_1 = -r_2 = -k_0 e^{-\frac{E_a}{RT}} c_1 \quad ; \quad Q = \kappa A (T - T_j)$$

The concentration, c, and reaction velocity, r, are column vectors with the length of two, to describe the components A and B.

The flow through the valves can be modelled as nonlinear functions of the control signals.

$$q = a\sqrt{2\Delta p} u^2 \quad ; \quad u \geq 0$$

This way of describing the problem is equation-oriented. Model representations, like this, do not have any structure. The model can not be reused in a new application. It is not easy to change the model and it is hard to read and understand the model for a unexperienced user.

3. Object-oriented Modelling

In object-oriented modelling models are represented as objects. Object-oriented modelling is based on the methodology from object-oriented programming. A good introduction to object-oriented methodology is given in Stefik and Bobrow (1984).

Model objects are represented as classes with attributs. The attributes can be other model objects. An object is a subclass of its super-class, which means that the subclass inherits properties from its super-class.

This model representation is implemented in the SEE prototype (Andersson, 1989a). A specification language for the model representation is called Omola, Object-oriented Modelling Language (Andersson, 1989b).

In this section we are first discussing some *model structuring concepts* and then the *inheritance concept*. Model decomposition, parameterization and inheritance are the basic elements in this modelling methodology.

Internal Model Structure

A model object have an internal structure of model component objects. The internal structure of a model is composed of three major component types, namely behaviour descriptions, terminals and parameters.

- *Behaviour description* or realization is a description of the model behaviour. The behaviour can be primitive, expressing the behavior symbolically with equations, or it can be composite and described with a structure of submodels. Models can have multiple realizations.
- *Terminal* is a model component which can be used to describe interaction with other components .
- *Parameter* is a model component that allow the user to interact with the model, in order to adapt its behaviour to new applications.

Model structuring concepts are disussed in more detail by Mattsson (1988). Model structures are also discussed in Åström and Kreutzer (1986) and in Åström and Mattsson (1987).

An object-oriented model representation of the reactor tank can be seen in Figure 2. It is composed of parameters, primitive behaviour description and terminals. There are five parameters and five terminals. The behaviour is described as dynamic mass and energy balances, which are the same as the first three differential equations in Section 2.

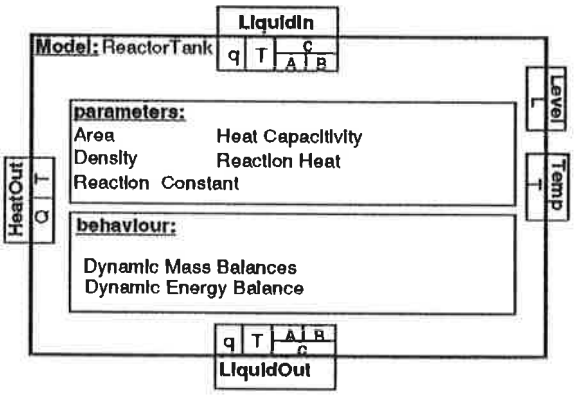

Figure 2. The internal structure of the reactor tank model.

Submodel Interaction

The structure of the reactor system is shown in Figure 3. The reactor system is described as a composite model object. The different parts are modularized into submodel objects. The connections between submodels represents submodel interactions. The interaction between two submodels is described by the terminal descriptions. If a connection is drawn between two terminals then the system make a consistency check. The two terminals must have the sane structure.

In this example the terminals are chosen so that the connections have natural interpretations. The connection between the tank reactor model, `TankReactor`, and the `Valve1` model

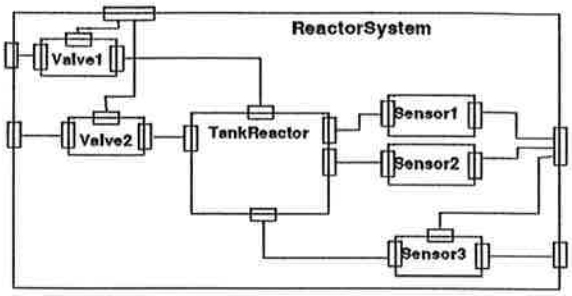

Figure 3. A block diagram showing the composite model of the tank reactor system.

object, see in Figure 3, have the physical interpretation of a pipe. In a mathematical model this connection means a set of relations between variables. The variables in this example is flow (q), temperature (T) and concentration, (c). An example of a structured terminal, **LiquidIn**, can be seen in Figure 2. Terminals describing interactions like this are common in process models. Terminals are defined as objects. This means that an process pipe terminal class can be a super-class of every process pipe terminal object in the process model. Terminal descriptions are therefore easy and natural to reuse. This kind of submodel interaction is well documented by Mattsson (1989).

The important result is the capability to encapsulate models. The terminal concept is the only way to describe interaction between submodel. This is a restriction but it guarantees the possibility to reuse models.

Hierarchical Submodel Decomposition

The tank reactor model can be decomposed into three submodels, namely one reactor vessel model (`ReactorTank`), one cooling jacket model (`Jacket`) and one heat transfer model (`HT-model`). The reactor vessel model is a primitive model and is seen in Figure 2. This means that the tank reactor model is a composite model, with three submodels, and we get a hierarchy of models. This is a hierarchical submodel description. This is shown in Figure 4.

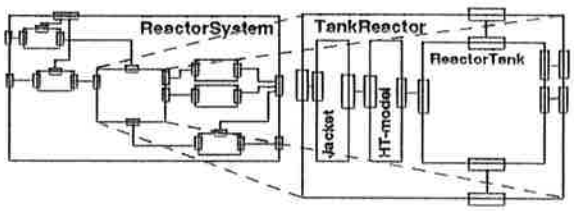

Figure 4. The hierarchical submodel decomposition in the tank reactor example.

This decomposition makes it possible to reuse submodels that is not directly interpretated as physical components. A heat transfer model

object is an example of this. It is now possible to change the heat transfer model without changing surrounding submodels or the super-model structure.

Parameterization

Parameterization of models and model components are important in our attempt to reuse objects.

A first parameterization is seen in Figure 2. Time invariant variables, like area and density, are defined as parameters.

The next step is structure parameterization. In the reactor tank model this means that the dimension of vectors, like concentration, are being set by parameters, which means that the reactor tank can be used in a new application with another number of chemical components.

A third method is to decompose the reactor tank model in one tank machine model and one chemical medium model. The machine model contains the main behaviour description (balance equations) and machine parameters (area). The medium model contains the medium behaviour, like reaction velocity description, and medium parameters, like reaction heat and density. This method is called *machine and medium decomposition*. The two submodels communicate through terminals. This way of decompose models can be seen as an advanced way of parameterization. This is an important method in chemical process applications.

Inheritance

Models are represented as objects, which are subclasses of predefined super-classes. A subclass inherits properties from its super-class. The model representation has single inheritance, which means that a subclass only has one super-class. The properties that are inherited are the object attributes, which are definitions of components. Model object inherits model component definitions.

A system defined super-class `Model` is the root of the model class hierarchy tree. A specialization of `Model` means that attributes defining model components are added to the subclass.

An example of how to use the inheritance concept is shown in Figure 5. The class `Valve` is a subclass of the system defined super-class `Model`. It is specialized by getting two attributes that define two model components. These model components are two terminals describing the inflow and the outflow of the valve object. `Valve` is a super-class to `ControlValve`, which have two additional attributes describing the control

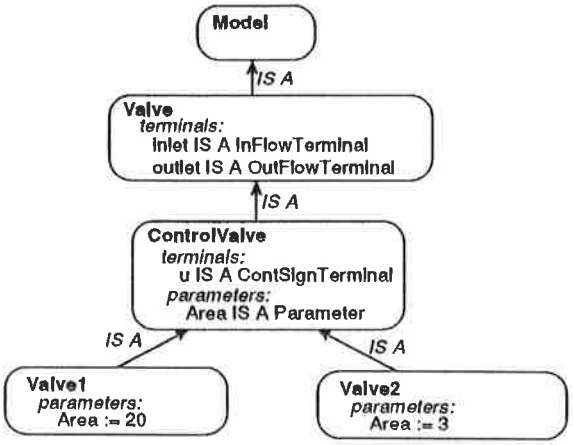

Figure 5. A part of model class hierarchy tree describing the relation between some valve models.

signal terminal and a parameter. The two valves used in the reactor system are specializations of `ControlValve`. They contain specializations of the parameter attribute `Area`.

Inheritance makes it possible to reuse previously defined models and model components and change them in desired ways. This means that we have two different hierarchies in object-oriented modelling. The first is the hierarchical submodel description and the second is the model class hierarchy. In object-oriented terminology they are called "PART OF" hierarchy and "IS A" hierarchy.

4. The Controlled Tank Reactor

We have seen how one can use an object-oriented approach to the modelling of the tank reactor process. Modularization, decomposition, parameterization and inheritance are the key words for this methodology.

The control system for the tank reactor process can be described in a similar way. The reactor has one control signal terminal that is connected to the two valves. It has also a measurement terminal describing the three sensors, level, temperature and outflow.

A super-model of the control system is connected to the reactor system through the control signal terminal and the sensor measurement terminal. This is seen in Figure 6. The control system is a composite model with submodels that represents the different controllers. In this example the terminals are chosen to fit the control system design and therefore are the control signals and measurement signals clustered into two structured terminals.

The first control system design is based on two PID-controllers, which is seen in Figure 6.

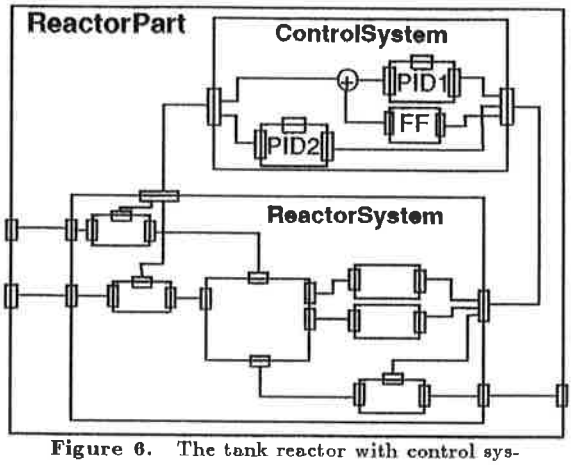

Figure 6. The tank reactor with control system.

One controller (PID1) uses the inflow valve to control the reactor level. The outflow measurement is used for feed forward control of the level. The other PID (PID2) controls the reactor temperature through the cooling medium valve.

Figure 7 illustrate the controller class hierarchy. The PID controllers, PID1 and PID2, are subclasses of a super-class describing the general PID controller, which is a subclass of a pole placement (PP) controller super-class. The general pole placement controller can be specialized to a PID through assignment of the polynomial degree parameters and polynomial structure description. The pole placement controller is a subclass of SISO, a controller class describing the structure of single input, single output controllers.

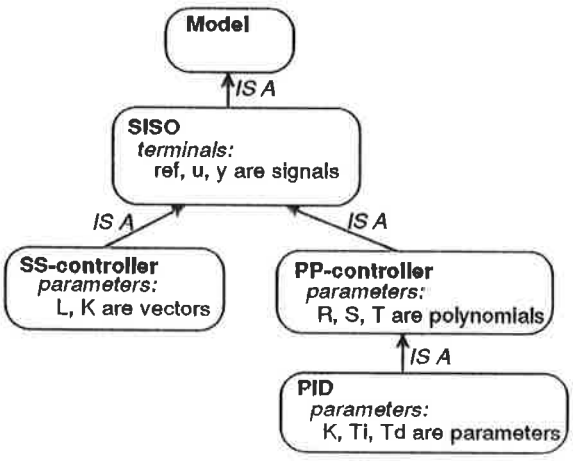

Figure 7. The model class hierarchy of controllers.

A new control design with another type of SISO controller can be introduced in the control configuration without any major changes. The attribute, where the PID controller definitions are done, can be overwritten by the name of the new controllers. In this case we can use state-space (SS) controller or pole placement controller because they are subclasses of SISO that describe the submodel interaction.

The structure of the control system can be changed by rewriting the behaviour description in the control system model object. This makes it possible to introduce a MIMO control design without any change of the ReactorPart model and the TankReactor model.

The resulting composite model, ReactorPart, can now be reused in a chemical process plant model. One example is the Process1 shown in Figure 8.

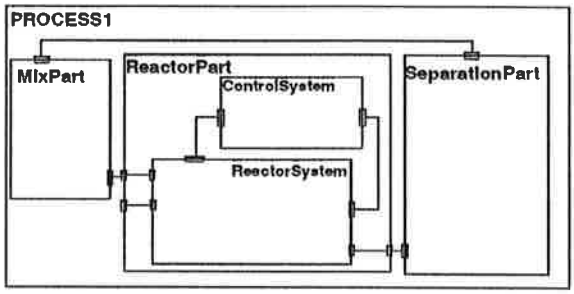

Figure 8. A chemical plant model that reuse the reactor process part model.

We have seen that it is possible to reuse controllers and control systems if we use modelling methodology in a proper way.

5. Discussions

Reuse, development, refinement and maintenance of models are facilitated through modularization, decomposition, parameterization and inheritance.

Model Reuse

How models and model components can be reused have been shown a number of times above. The strong modularization concept with encapsulated submodels with terminals makes it possible to reuse submodels. Submodels can be stored in model libraries. Decomposition of models into submodels makes it possible to reuse the structure and change the submodels in the structure. Advanced parameterization of models can increase the reusability of model. Inheritance means that the model object description can be distributed in a tree of super-classes and are therefore reused.

Model Development

Model development is facilitated in two ways. One is the possibility to reuse submodels from model libraries. Predefined submodels can be

reused in new composite models describing new applications. This is possible due to the strong modularization. One example is to use common process equipments, like pumps, valves etc., to create a complex process.

Another way to develop models is to use the inheritance and specialize predefined objects to describe new applications. This way to develop models is of major importance and has a great potential. This is shown both in the tank reactor example and in the control system example.

Model Refinement

A model class can have multiple realizations. This not discussed in the paper. This means that a model can have more then one behaviour descriptions. This can be used to refine the behaviour of models. A model can first get a simple behaviour description. It can then easily be refined by getting an additional behaviour description. The behaviour descriptions can be static, dynamic, simple, complex, linear or nonlinear. The user can choose a desired realization depending on the application.

Model Maintenance

Long term use of process models requires possibilities to reuse and refine models. The model reuse and model refinement have already been discussed above. To use models for different purposes requires new tools that can operate on the model. Examples are tools for steady-state calculation, optimization, linearization and control design. The SEE architecture allows introduction of new tools like these. The model representation with multiple realizations make it possible to use models in different tools and for different purposes.

Model Presentations and Libraries

All the desired properties are results of well designed modularization decomposition and parameterization of models. The result is large distributed model libraries. A classical problem then arise. How should the model library be organized and how should the user be able to find what he needs? I think this is one of the key issues in object-oriented modelling.

6. Conclusions

Object-oriented modelling facilitates reuse, development, refinement and maintenance of models. These properties are important for applications of models in process industry. The main argument is to decrease the investment in process model development and process model maintenance. The key elements to create these properties are modularization, decomposition, parameterization and inheritance.

7. Acknowledgements

I would like to thank Karl Johan Åström for initializing this work. I would also like to thank Mats Andersson and Sven Erik Mattsson, who are behind the object-oriented modelling architecture, for many and long useful discussions.

8. References

ANDERSSON, M. (1989a): "An Object-Oriented Modelling Environment," *1989 European Simulation Multiconferance, ESM'89, Rome, Italy.*

ANDERSSON, M. (1989b): "Omola - An Object-Oriented Modelling Language," Report TFRT-7417, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

ÅSTRÖM, K.J. and W. KREUTZER (1986): "System Representations," *IEEE Third Symposium on Computer-Aided Control System Design, Arlington, Virginia.*

ÅSTRÖM, K.J. and S.E. MATTSSON (1987): "High-Level Problem Solving Languages for Computer Aided Control Engineering," Report TFRT-3187, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden.

MATTSSON, S.E. (1988): "On Model Structuring Concepts," *4th IFAC Symposium on Computer-Aided Design in Control Systems, P.R. China.*

MATTSSON, S.E. (1989): "Modeling of Interactions between Submodels," *1989 European Simulation Multiconferance, ESM'89, Rome, Italy.*

NILSSON, B, S.E. MATTSSON and M. ANDERSSON (1989): "Tools for Model Development and Simulation," *Proceedings of the SAIS '89 Workshop.*

STEFIK, M. and D.G. BOBROW (1984): "Object-Oriented Programming: Themes and Variations," *The AI Magazine, Vol. 6, No. 4.*